Generating Paraphrases with Lean Vocabulary

Tadashi Nomoto
National Institute of Japanese Literature
10-3 Midori Tachikawa Tokyo, 190-0014, Japan
nomoto@acm.org

Abstract

In this work, we examine whether it is possible to achieve the state of the art performance in paraphrase generation with reduced vocabulary. Our approach consists of building a convolution to sequence model (Conv2Seq) partially guided by the reinforcement learning, and training it on the subword representation of the input. The experiment on the Quora dataset, which contains over 140,000 pairs of sentences and corresponding paraphrases, found that with less than 1,000 token types, we were able to achieve performance that exceeded that of the current state of the art.

1 Introduction and Background

The past few years have seen the surge of interest among NLP researchers in applying deep learning to the end-to-end generation of paraphrases: where the goal is to create a sentence semantically close to but distinct in style from the source sentence, without a recourse to some linguistically motivated devices as was often the case with the work a decade ago (Barzilay and Lee, 2003; Power and Scott, 2005; Duboué and Chu-Carroll, 2006; Zhao et al., 2009).

Among the more recent efforts that are of relevance to this work are (Gupta et al., 2017; Iyyer et al., 2018; Wieting et al., 2017; Li et al., 2018), each of which employs the deep learning one way or another to address particular aspects of paraphrase generation. (Iyyer et al., 2018) was concerned with generating paraphrases in a controlled fashion so that the outputs adhere to a certain syntactic requirement specified by the user. (Wieting et al., 2017), meanwhile, worked on creating a high quality paraphrase corpus through back-translation and argued that they brought the quality of paraphrases close to that of those manually written. (Gupta et al., 2017) was the first attempt to bring the variational auto-encoder (VAE), wildly popular in computer vision, to bear on the current issue. The idea is to cast paraphrase generation as a problem of sampling a hidden encoding from VAE that retains some similarity to the input sentence. (Li et al., 2018) is a paper most closely related to this one. It features Monte Carlo policy gradient and Inverse Reinforcement Learning, which achieved record setting results on the Quora dataset. Broadly, this paper falls in line with (Li et al., 2018), except that we aim at pushing their results even further by letting reinforcement learning work with the lean vocabulary enabled by the subword encoding.

2 Approach

While we work with a convolution to sequence architecture as provided by Facebook at Github (Fairseq),[1] we made a few changes to the criterion (loss) part of the model in order to accommodate the policy gradient variety of reinforcement learning (PRL). What it does is to allow the model to *randomly* explore the vocabulary space for each decoded output of the model, in search of a word sequence that results in a higher BLEU score. The goal of PRL is to nudge the model toward words that lead to a higher cumulative gain over a long run in terms of BLEU, not just those that result in an immediate reward in the form of the cross entropy. Formally, PRL aims to maximize the following quantity:

$$U(\theta) = \log \pi(\tau; \theta) R(\tau)$$

i.e., an expected return from taking a trajectory τ (word sequence) under π, a policy parameterized with θ. Note that a policy here is understood as a stochastic function that indicates how

[1] In particular, one called 'fconv_iwslt_de_en,' which is part of fconv.py found at https://github.com/pytorch/fairseq

likely a particular word occurs following the previous one, which is equivalent to a language model $P(w_t \mid w_0, \ldots, w_{t-1})$. $R(\tau)$ is a reward function that determines how much reward will be given for a particular path pursued.

Formally, R is defined as:

$$R(\tau) = \sum_t^T R_t(w_t) \quad \text{where:}$$
$$R_t(w_t) = r_t + \sum_{i=t+1}^{T-1} \lambda R_i(w_i)$$

t represents a time step at which each word appears in the trajectory of length T. r_t represents a reward for the entire trajectory τ against the target, and λ a decay coefficient, which is set at 1.0 for our case. Note that $U(\theta)$ amounts to an expected reward under the policy π. Importantly, we do not use a baseline for R, as we did not find it much of use in our experiment, which is consonant with (Wu et al., 2018), who reported that its use led to no improvement in neural machine translation. Crucially, however, we will make use of multiple reinforcement objectives: a bigram based BLEU and **seqratio**, a string similarity metric based on Levenshtein distance.[2]

The loss function for the present model now looks like:

$$\mathcal{L} = \mathcal{V}(f(u), t) + R(\tau) * \mathcal{V}(f(u), \tau)$$

where $\mathcal{V}$ is a cross-entropy loss, $f(u)$ is an output of Conv2Seq applied to the input u, and t a target. In training, R *alternates* between BLEU and **seqratio**[3] as it moves from one mini-batch to another. Of note is that we have not employed any particular scheduling tactic for training, in contrast to (Ranzato et al., 2015), who initially trained the model on the ground truth and let it slowly deviate from the optimal policy. More in line with the present approach is (Paulus et al., 2017), who worked with a mixed learning objective which attempts to maximize the probability of generating a target given the input, while searching for an optimal policy, except that they rewarded actions that would encourage the generation of fluent text, not those that might improve an evaluation metric such as BLEU.

Another important idea that underlies the current approach is the notion of *subword*, whose members include anything from a single letter or symbol to a full-fledged word. When reviewing errors incurred in an earlier experiment, we came to realize that the majority of them were caused by the model's inability to handle out-of-vocabulary words (OOVs). The observation led us to explore the use of subword as a way to represent text, which turned out to be quite effective, reducing OOVs, and boosting the over-all performance. A particular approach we adopt here is one by (Kudo, 2018). It works by replacing a sentence with a string of reasonably likely textual fragments (or subwords), which may or may not correspond to a meaningful linguistic expression. Under the subword scheme, OOVs are less likely to occur because any word that is not part of the vocabulary will be broken up into or replaced with those that are part of the dictionary. If anything, one can revert to representing a word as a string of alphabets which are always kept in the dictionary (Kudo, 2018).

To recap, two key ideas that drive the current approach are: the adoption of a learning objective informed by multiple reinforcement objectives in addition to the ground truth and the use of subwords to ensure that the model knows every word it sees in the input. For the sake of convenience, we refer to the present approach as RIPER (Reduced InPut Model Enhanced with Reinforcement Learning).

3 Experiment Setup

In the experiment, we made use of the Quora duplicate dataset (QRA), which contains 404,302 pairs of questions and their near duplicates that appeared on the Quora's online forum,[4] out of which we extracted some 140,000 pairs marked as a duplicate for use in training and testing. We follow the same setup as one used in (Li et al., 2018), which involves dividing the corpus into 100K for training, 3K for validation and 30K for testing. In addition to QRA, we tested the approach on the ParaNMT-50M dataset (Wieting and Gimpel, 2017) (PMT).[5] PMT is a fairly large corpus built by back-translating what is translated into Czech with machine translation, into English again using machine translation. We report all the results

[2] `https://github.com/ztane/python-Levenshtein.git`

[3] It is the averaged sum of lengths of common substrings, with the range of 0 (least similar) to 1 (most similar).

[4] `https://www.kaggle.com/c/quora-question-pairs`

[5] The corpus we used here contains paraphrase pairs randomly sampled from one of its variants called Para-nmt-5m-processed where staggering 5,370,128 pairs of sentences are found.

Table 1: Corpus Statistics (in sentences)

	TRAIN	VALID	TEST
PMT	600,000	5,000	5,000
QRA	100,000	3,000	30,000

Table 2: Results on PMT and QRA in BLEU.

VOC.	MODEL	PMT	QRA
subword	V8K	**0.414**	0.447
	V8K$^-$	0.411	0.444
	V4K	0.412	0.448
	V4K$^-$	0.397	0.448
	V1K	0.392	**0.453**
	V1K$^-$	0.391	0.442
word	BSE$^+$	0.312	0.295
	BSE	0.395	0.406
	VAE	0.154	0.295
	R-RL	–	0.418
	NMT	0.328	0.399
	RbM-SL	–	0.435
	RbM-IRL	–	0.431

in the bigram based BLEU which we used for reinforcement learning. We also looked at how the present framework would fare against some of the prior approaches tackling the same problem. In this work, we are focusing on some of the more recent attempts, VAE-SVG-eq (Gupta et al., 2017), RbM-SL and its variant, RbM-IRL (Li et al., 2018), along with other more conventional approaches. The former is based on the variational auto-encoder (Kingma and Welling, 2013) while RbM-SL draws essentially on Monte Carlo Policy Gradient, enhanced with reward reshaping (Ng et al., 1999). RbM-IRL differs from RbM-SL in that it makes use of *inverse reinforcement learning* whose goal is to reconstruct a reward function by directly exposing the model to the optimal state/action transition taken by human.

We report results for VAE-SVG-eq from running the third party implementation.[6] As for the latter models (RbM-SL/IRL). we repeat what was published as there is no code available to reproduce runs. We note that for the subword encoding, we have taken advantage of the code published as a part of (Kudo, 2018) on Github.[7]

[6]`https://github.com/ale3otik/paraphrases-generator`

[7]`https://github.com/google/sentencepiece`

4 Results and Discussion

Table 2 summarizes major results. To start, V8K, V4K and V1K are all RIPER models that feed on subwords with dictionaries of sizes 8,000, 4,000, and 1,000, respectively. Note that all of them are derived from the Conv2Seq model, which we call BSE and serves as a baseline.[8] To see the effect of reinforcement, we created some variations on RIPER which differed from the originals in that they were decoupled from the reinforcement capability. To highlight the difference, we mark them with a minus sign. Thus 'V1K$^-$' denotes a V1K with the reinforcement removed. Similarly for V4K and V8K. Furthermore, we had another model, created by extending BSE with the reinforcement, denoted here by BSE$^+$, which works at the word level. VAE is a third party implementation of VAE-SVG-eq. NMT is another baseline derived from OpenNMT-py, a sequence to sequence machine translation model,[9] which we used with the default settings.[10] R-RL corresponds to a pointer generator network empowered with reinforcement learning based on ROUGE (Ranzato et al., 2015). We took its score on QRA from (Li et al., 2018). VOC. indicates the kind of vocabulary the models use: 'word' means that they use tokens as identified by a tokenizer while 'subword' indicates that they work on word fragments as produced by (Kudo, 2018). In the following, we take RbM-SL as the state of the art (SOTA).

In QRA, we see that RIPERs are doing well across the board, beating all the rest, which includes the SOTA. It is remarkable that V1K outruns others, with as little as 1K tokens. On PMT, V8K comes out a winner. The fact that RIPERs did well on two reasonably large datasets and even outperformed the SOTA, drives home the utility of vocabulary reduction through the subword trick. The results also show that reinforcement learning benefitted to some degree from the reduced vocabulary, a point underscored by the fact that BSE$^+$ fell visibly behind BSE both on PMT and on QRA, which can only be explained as the result of having a large action space it had to deal with. By contrast, the use of subwords, with the concomitant reduction in search space, led V1K to reach

[8]Refer to footnote 1 for details

[9]`https://github.com/OpenNMT/OpenNMT-py`

[10]In particular, we enabled options related to the copying functionality, `-copy_attn` and `-reuse_copy_attn`

Table 3: Generated Paraphrases (V8K/PMT)

source	target	generated
she staggered , went to one knee , and was up again almost at once .	she staggered , dropped to her knee , but she got up again .	she staggered , went to one knee and stood up again .
i stopped by your room , you know , because i recognized your name .	yes , i was in your room because i know your name and ...	i stopped by your room because i recognized your name .
moving to his command post , he barked , “ get me visual ! ”	he headed for his command post and barked , “ let go of the picture ! ”	he ’s moving to his command post , he barked , “ get me visual ! ”
at the point when we were developing this , nobody knew that there were drugs over there .	by the time we torched it , no one knew there were drugs in it .	at the point when we got this , no one knew there were drugs .

Table 4: Generated Paraphrases (V1K/QRA)

source	target	generated
why there is a ban of 500 and 1000 notes ?	what are the pros and cons of banning 500 and 1000 rupee notes in india ?	what are the ill effects of demonetization of 500 and 1000 rupees notes ?
what is ultimate purpose of life ?	what purpose do you find in life ?	what is the purpose of life if not money ?
what can i do to improve my english speaking ?	how can i improve my english pronunciation ?	how can i improve my english pronunciation ?
what is the best laptop in 50000 in india ?	what could be the best laptop in budget upto 50k ?	which is the best laptop to buy under rs 50000 ?
why were the 500 and 1000 rupee notes demonetized ?	what is the logic behind scrapping of 500 and 1000 rupee notes by the indian government ?	why did the indian government demonetize the current 500 and 1000 rupee notes ?

Table 5: The number of tokens in the vocabulary

	PMT	QRA
V8K	7,981 (4.8%)	7,992 (27.9%)
V4K	3,983 (2.3%)	3,994 (14.0%)
V1K	994 (0.5%)	995 (3.5%)
BSE	166,899	28,615

as high as 0.453 on QRA while leaving behind its word based cousin at 0.295. Across the results, there is a general tendency for the subword based models to improve through reinforcement, while those dependent on words suffer the steep decline.

Table 5 shows how much reduction is achieved through the subwording. V1K on QRA works with less than 4% of the vocabulary used by BSE, and on PMT as little as 0.5%. One thing we should point out is that on PMT, we have the maximum score with V4K, while on QRA, the best result comes from V1K. Its performance appears to peak when run on 3 to 5% of the vocabulary. The question of how broadly the observation holds is an interesting one, though it is not clear what the answer would be at this point.

We conclude the section by leaving the reader with some paraphrases RIPERs generated (Table 4), to give an intuitive sense of how we are doing.

5 Conclusions

In this paper, we introduced a fairly straightforward yet novel approach to paraphrase generation we call RIPER, which leverages a recent development in the subword encoding and a policy gradient variety of reinforcement learning. The experiments with two large datasets PMT and QRA have shown unequivocally that RIPER works. One key takeaway is the idea of coupling the subword encoding with reinforcement learning (RL). The application of RL to NLP has been generally perceived as a difficult one (Irpan, 2018). One reason is the enormity of search space due to ever growing vocabulary the model works with. The subword encoding is a nice trick which may mitigate the problem, and could lead to a wider adoption of RL in NLP.

6 Acknowledgement

The present work was partly funded by the Microsoft Core Fellowship for 2018-2019.

References

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. *CoRR*, cs.CL/0304006.

Pablo Ariel Duboué and Jennifer Chu-Carroll. 2006. Answering the question you wish they had asked: The impact of paraphrasing for question answering. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 4-9, 2006, New York, New York, USA*.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2017. A deep generative framework for paraphrase generation. *CoRR*, abs/1709.05074.

Alex Irpan. 2018. Deep reinforcement learning doesn't work yet. `https://www.alexirpan.com/2018/02/14/rl-hard.html`.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. *CoRR*, abs/1804.06059.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *CoRR*, https://arxiv.org/abs/1312.6114.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. *CoRR*, abs/1804.10959.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018. Paraphrase generation with deep reinforcement learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3865–3878. Association for Computational Linguistics.

Andrew Y. Ng, Daishi Harada, and Stuart J. Russel. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML '99 Proceedings of the Sixteenth International Conference on Machine Learning*, pages 278–287. Morgan Kaufmann.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *CoRR*, abs/1705.04304.

Richard Power and Donia Scott. 2005. Automatic generation of large-scale paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *CoRR*, abs/1511.06732.

John Wieting and Kevin Gimpel. 2017. Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. *CoRR*, abs/1711.05732.

John Wieting, Jonathan Mallinson, and Kevin Gimpel. 2017. Learning paraphrastic sentence embeddings from back-translated bitext. *CoRR*, abs/1706.01847.

Lijun Wu, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018. A study of reinforcement learning for neural machine translation. *CoRR*, abs/1808.08866.

Shiqi Zhao, Xiang Lan, Ting Liu, and Sheng Li. 2009. Application-driven statistical paraphrase generation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 834–842, Stroudsburg, PA, USA. Association for Computational Linguistics.